C++ ASSIGNMENT

1.Ques :Given a linked list, swap every two adjacent nodes and return its head. You must solve the problem without modifying the values in the list's nodes (i.e., only nodes themselves may be changed.)

Ans:

```
class Solution {
public:
  ListNode* swapPairs(ListNode* head) {
    if(head==NULL || head->next==NULL) return head;
    ListNode* dummy=new ListNode(0);
    dummy->next=head;
    ListNode* td=dummy;
    while(td->next && td->next->next){
      ListNode* swap1=td->next;
      ListNode* swap2=td->next->next;
      swap1->next=swap2->next;
      swap2->next=swap1;
      td->next=swap2;
      td=swap1;
    }
    return dummy->next;
  }
};
```

2.Ques :You are given the head of a linked list, which contains a series of integers separated by 0 's. The beginning and end of the linked list will have Node.val == 0 .For every two consecutive 0 's, merge all the nodes lying in between them into a single node

whose value is the sum of all the merged nodes. The modified list should not contain any 0 's.Return the head of the modified linked list.

Ans:

```
class Solution {
public:
  ListNode* mergeNodes(ListNode* head) {
        ListNode* zero=head;
        ListNode* p=head->next;
        while(p){
          if(p->val!=0){
            zero->val+=p->val;
          }
          else{
            if(p->next==NULL) zero->next=NULL;
            else{
              zero->next=p;
              zero=p;
              zero->val=0;
            }
          }
          p=p->next;
        }
        return head;
  }
};
```